

(19) 日本国特許庁 (JP)

## (12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平 10 - 289114

(43) 公開日 平成10年(1998)10月27日

(51) Int. Cl.<sup>6</sup>

G 0 6 F

9/46

識別記号

3 4 0

F I

G 0 6 F

9/46

3 4 0 B

3 4 0 E

審査請求 未請求 請求項の数 4

O L

(全 1 4 頁)

(21) 出願番号

特願平9-96263

(22) 出願日

平成9年(1997)4月14日

(71) 出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72) 発明者 鈴木 定佳

東京都品川区北品川6丁目7番35号 ソニー  
株式会社内

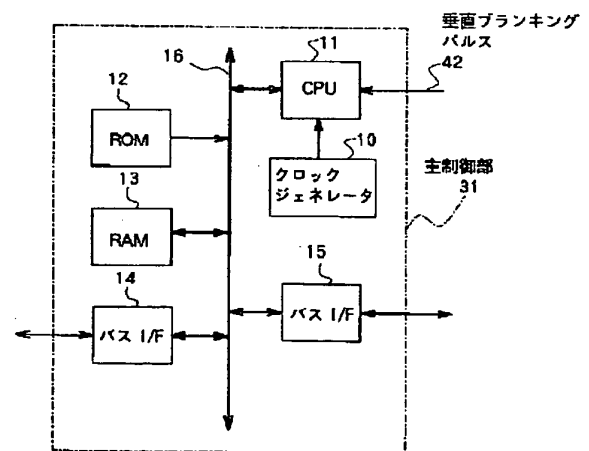
(74) 代理人 弁理士 藤島 洋一郎

(54) 【発明の名称】 マルチタスク制御方法、マルチタスク制御プログラムを記録した記録媒体、およびマルチタスク制御プログラムを組み込んだ電子機器

(57) 【要約】

【課題】 所要のメモリサイズを削減できるマルチタスク制御方法、およびマルチタスク制御プログラムを記録した記録媒体、ならびにこのマルチタスク制御プログラムを組み込んだ電子機器を提供する。

【解決手段】 タスクBとタスクCとが共にレディ状態にあって起動要求が競合したときは、原則通り、優先順位に従って、より高い優先順位のタスクに対してCPUの占有権が与えられる。タスクBまたはCの実行中にタスクAから割り込みがあったときは、直ちにCPUの占有権をタスクAに与える一方、タスクB（またはC）の実行中にタスクC（またはB）からの割り込み要求があったときは、そのタスクの優先順位の如何に係わらず、その実行中のタスクの終了を待ってCPUの占有権を開放する。必要なタスク処理のみがリアルタイム動作し、さほどこリアルタイム性を必要としないタスクは、擬似的なリアルタイム動作をする。



## 【特許請求の範囲】

【請求項1】 電子機器の制御上必要な個々の機能に対応して設けられた複数のタスクを制御するための方法であって、

前記複数のタスクのうち、前記電子機器を制御する上で即時実行の要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、

前記特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたことを特徴とするマルチタスク制御方法。

【請求項2】 前記特定のタスクの割込みによって実行が中断されたタスクは、その特定のタスクが実行を終了した時点において、優先的に実行を再開することを特徴とする請求項1記載のマルチタスク制御方法。

【請求項3】 電子機器の制御上必要な個々の機能に対応して設けられた複数のタスクを制御するためのマルチタスク制御プログラムであって、

前記複数のタスクのうち、前記電子機器を制御する上で即時実行が要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、前記特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたマルチタスク制御プログラムを記録した記録媒体。

【請求項4】 電子機器の制御上必要な個々の機能に対応して設けられた複数のタスクを制御するに際し、前記複数のタスクのうち、前記電子機器を制御する上で即時実行が要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、前記特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたマルチタスク制御プログラムを組み込んだ電子機器。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】 本発明は、電子機器の制御上必要な個々の機能に対応したプログラムとして設けられた複数のタスクを制御するためのマルチタスク制御方法、およびマルチタスク制御プログラムを記録した記録媒体、ならびにこのマルチタスク制御プログラムを組み込んだ電子機器に関する。

## 【0002】

【従来の技術】 従来より、例えば、一般家庭用または業務用のテレビジョン受像機やコンピュータ用ディスプレイ装置等の映像表示装置においては、操作キーから入力されたデータの処理をするタスク、画面表示の制御を担

当するタスク、あるいは入力される映像・音声信号における同期信号の周波数測定を担当するタスク等、映像表示装置が有する各機能ごとにタスクをそれぞれ独立に作成し、これらのタスクをCPUがOSに従って統括管理するようになっている。このように複数のタスクに機能を分担して割り当てることによって、多人数によるプログラムの並行開発が可能となる等、プログラムの開発効率を向上させることができる。

【0003】 このような複数のタスクを管理するマルチタスク制御型のOSにおいては、それぞれのタスク処理をリアルタイムに実行できることが理想であり、このリアルタイム処理を実現するため、従来より各種のリアルタイムOS(RTOS)が提案されている。

【0004】 図15は、従来のリアルタイムOSによるマルチタスク制御方法を表すものである。この例では、3つのタスクA～CによるCPUの占有状態をOSが制御するものとし、このうちタスクAの優先順位が最も高く(P=1)、タスクBの優先順位が次に高く(P=2)、タスクCの優先順位がもっとも低い(P=3)ものとする。

【0005】 図15に示したように、当初、最下位優先順位のタスクCが実行状態であったとする。この状態で、時刻τ1においてタスクBの起動要求があると、OSはタスクCによるCPUの占有を一旦中断させ、タスクBにCPUの占有権を渡す。このとき、タスクCは、図16に示したように、中断時におけるCPUの状態を格納した各種レジスタの内容をRAM(Random Access Memory)内に確保したタスクC用スタック領域に退避(プッシュ)してその後の処理の再開に備える。さらに、時間が経過し、時刻τ2において、優先順位が最も高いタスクAの起動要求があると、OSはタスクBによるCPUの占有を一旦中断させ、タスクAにCPUの占有権を渡す。このとき、タスクBは、中断時におけるCPUの状態を格納した各種レジスタの内容をRAM内のタスクB用スタック領域(図16)に退避(プッシュ)してその後の処理の再開に備える。さらに、時間が経過して、時刻τ3においてタスクAの実行が終了すると、タスクAはCPUの占有を開放する。この時点では、タスクBおよびタスクCが再開待ち状態になっているが、タスクBの方がタスクCよりも優先順位が高いので、OSは、CPUの占有権をタスクBに与える。このとき、タスクB用スタック領域(図16)に退避されていた各レジスタデータがCPU内のレジスタ群にそれぞれ戻され(ポップされ)、中断時のCPUの状態が復元されて、タスクBの実行が再開される。さらに、時間が経過して、時刻τ4においてタスクBの実行が終了すると、タスクBはCPUの占有を開放する。この時点では、タスクCのみが再開待ち状態になっているので、OSは、CPUの占有権をタスクCに与える。このとき、タスクC用スタック領域(図16)に退避されていた各レジ

タデータがCPU内のレジスタ群にそれぞれ戻され、中断時のCPUの状態が復元されて、タスクCの実行が再開される。なお、図16において、A用スタック領域は、タスクAの実行中に外部からの割り込みが生じた場合やサブルーチンコールを行った場合等に、その時点のCPUのレジスタ群の内容を一時退避するためのものである。

【0006】このように、従来は、各タスクに優先順位を割りつけると共に、あるタスクの実行中に優先順位の高い他のタスクからの割り込み要求があったときは、その実行中のタスクの処理を必ず中断してその割り込み要求をした上位タスクの実行を即時確保するようにしていた。このため、OSが管理するすべてのタスクについてそれぞれ専用のスタック領域を用意する必要があった。

【0007】

【発明が解決しようとする課題】ところで、このようなRTOSを用いて構築したシステムとして、TRON(The Realtime Operating system Nucleus)と呼ばれるコンピュータアーキテクチャ体系が知られているが、その中に、機器組込制御用マイクロコンピュータの標準OSを目指したμITRON(micro Industrial TRON; マイクロアイトロン)と呼ばれるものがある。このμITRONでは、汎用性を確保すべく、個々の電子機器からすれば不必要な機能についてまでリアルタイム性を追求している。例えば、OSの管理下にあるすべてのタスクに対してリアルタイム動作が要求されていたり、不必要なシステムコールが行われる等、過剰な仕様となっている場合が多い。具体的には、ある電子機器からみれば不要なタスク状態管理用フラグが存在していたり、あるいは不要なプログラムが含まれていること等がある。ここで、通常、各タスクはプログラムとしてROM(Read Only Memory)に格納されるが、上記のように不要な部分が多く含まれていると、限られた記憶容量しかもたないROMというハードウェア資源が無駄に消費されることとなり、このことがプログラム開発上で大きな障害の原因となる場合があった。

【0008】また、この種のOSでは、各タスクに平等にリアルタイム処理を保証する必要があることから、図15において説明したように、タスクA~Cのそれぞれについて専用のスタック領域を用意する必要がある。このため、必要なスタックメモリ容量は、タスク数に応じて増加することになる。

【0009】そこで、必要となるスタックメモリの容量を削減すべく、スタックメモリを共有化することが考えられるが、この場合には、その共有化のためのプログラム等を別途OSに組み込む必要が生じ、却ってROMの記憶容量を多く消費する結果となる。

【0010】本発明はかかる問題点に鑑みてなされたもので、その目的は、所要のメモリサイズを削減することができるマルチタスク制御方法、およびマルチタスク制

御プログラムを記録した記録媒体、ならびにこのマルチタスク制御プログラムを組み込んだ電子機器を提供することにある。

【0011】

【課題を解決するための手段】本発明に係るマルチタスク制御方法は、複数のタスクのうち、電子機器を制御する上で即時実行の要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたものである。ここで、特定のタスクの割り込みによって実行が中断されたタスクは、その特定のタスクが実行を終了した時点において、優先的に実行を再開するようにすることができる。

【0012】本発明に係るマルチタスク制御プログラムを記録した記録媒体は、複数のタスクのうち、電子機器を制御する上で即時実行が要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたマルチタスク制御プログラムを記録したものである。

【0013】本発明に係るマルチタスク制御プログラムを組み込んだ電子機器は、複数のタスクのうち、電子機器を制御する上で即時実行が要求される特定のタスクに対しては、そのタスクの実行を要求する命令に応じて即時そのタスクを起動させるリアルタイム制御を行う一方、特定のタスク以外のタスクに対しては、そのタスクの実行を要求する命令があった時点において実行状態となっている他のタスクの実行終了を待って起動させるようにしたマルチタスク制御プログラムを組み込んだものである。

【0014】本発明に係るマルチタスク制御方法では、即時実行が要求される特定のタスクに対してのみ限定的にリアルタイム制御が行われ、それ以外のタスクに対してはいわば擬似的なリアルタイム制御が行われる。

【0015】

【発明の実施の形態】以下、本発明の実施の形態について図面を参照して詳細に説明する。

【0016】図1は本発明の一実施の形態に係るマルチタスク制御方法が適用される電子機器としての映像表示装置の概略構成を表すものである。この映像表示装置は、主制御部31と、ユーザインタフェース(I/F)制御部32と、オンスクリーンディスプレイ(OSD)制御部33と、不揮発性メモリ34と、偏向制御回路35とを備え、シリアルバス30によって相互に接続され

ている。本装置はまた、 $I^2C$  (Inter Integrated Circuit) バス36によって主制御部31に接続されたビデオ制御回路37と、ビデオ制御回路37およびOSD制御部33に接続されたRGBデコーダ38と、このRGBデコーダ38の出力端側に接続されたCRT (陰極線管) 39とを備えている。

【0017】主制御部31は、映像表示装置全体の動作を制御するためのもので、CRT39における表示画面の各垂直帰線期間のタイミングごとに偏向制御回路35から入力される垂直ブランキングパルス (V-BLKPulse) 42に同期してバスアクセスを行い、全体の動作を制御するようになっている。具体的には、RGB入力端子からビデオ制御回路37に入力されたRGB信号 (R, G, Bの各色信号) 43の状態を制御したり、ユーザインタフェース制御部32を介し、図示しない音量キーやメニューキー等のパネル操作キーから入力され、またはリモートコントローラ (図示せず) から受光部を経て送られてきたコマンドデータ等に応じて、シリアルバス30に接続された各デバイスや図示しない電源供給状態表示用の電源LED (発光ダイオード) 等を

制御するようになっている。

【0018】ユーザインタフェース制御部32は、パネル操作キーや受光部等からの入力データをサンプリングし、主制御部31からの問い合わせに応じてそのサンプリングデータをシリアルバス30を介して送出したり、主制御部31からの要求に応じて電源LEDを点灯させる等の制御を行う。OSD制御部33は、主制御部31からの指示に基づいてCRT39に映像と重畳させて表示させるキャラクタを表すカラーデータをRGB信号41としてRGBデコーダ38に出力するようになっている。偏向制御回路35は、CRT39の水平および垂直ビームの偏向を制御するためのもので、この回路内のいくつかのデバイスがシリアルバス30を通して主制御部31から制御されるようになっている。この回路は、例えば水平同期周波数が31.5kHz、垂直同期周波数が70/60/50Hzという走査周波数で偏向制御を行う、いわゆるシングルスキャンシステムを構成している。ビデオ制御回路37は、 $I^2C$  バス36を通して主制御部31から与えられる指示に基づいて、外部RGB入力端子から入力されたRGB入力信号43に所定の信号処理を施し、RGB映像信号45としてRGBデコーダ38に入力するという制御を行う。RGBデコーダ38は、ビデオ制御回路37からのRGB映像信号45と、OSD制御部33からのRGB信号41とを混合してCRT39に送出するようになっている。

【0019】図2は、図1における主制御部31の概略構成を表すものである。この主制御部31は、クロックジェネレータ10から出力される所定の周波数のクロックに同期して動作するCPU11と、各種のプログラムを格納したROM12と、ワーク用メモリとしてのRA

M13と、 $I^2C$  バス36とのバス接続を行うバスインタフェース14と、シリアルバス30とのバス接続を行うバスインタフェース15と、これらを相互に接続する内部バス16とを備えている。

【0020】CPU11は、ALU (算術論理演算部)、インストラクションデコーダ、累算器 (アキュムレータ)、およびタイミング制御部等の各種回路のほか、インストラクションレジスタ、フラグレジスタ、プログラムカウンタ等の専用レジスタ、その他、汎用レジスタ等 (いずれも図示せず) を備えて構成されている。このCPU11には、図1の偏向制御回路35から一定周期で垂直ブランキングパルス42が入力されると共に、その内部では、後述するオペレーティングシステムが各タスクを制御する際の基準となるOS基準割込信号が一定周期で発生するようになっている。

【0021】ROM12は、オペレーティングシステム (OS) の本体部 (以下、単にOSという。) のほか、CPUが行うべき動作を機能ごとに分担するタスクプログラム (以下、単にタスクという。) 等を格納するためのものである。本実施の形態では、説明の簡単のため、図3に示したように、OSとかみ合う形で3つのタスクA, B, Cが組み込まれているものとする。後述するように、OSはタスクA~Cのそれぞれからの要求あるいは命令 (システムコール) に応じて、CPU11の占有権を各タスクに与える制御を行う。

【0022】タスクA~Cは、それぞれ、属性1または属性2のいずれかを有する。ここで、属性1のタスクは優先順位が最も高いタスクであり、上記した垂直ブランキングパルス42による割り込みがあった場合のシステムコールに応じて他のタスクの実行を中断させ、自らが即時実行状態となることが出来るタスクである。そして、この属性1のタスクは、必ずタスクのプログラムの最初から実行されるようになっている。本実施の形態では、タスクAのみが属性1に属するものとする。ここで、タスクAは、本発明における特定のタスクに対応し、タスクBおよびタスクCは本発明における特定のタスク以外のタスクに対応する。

【0023】一方、属性2のタスクは、優先順位が2位以下の任意の順位に設定可能なタスクであり、垂直ブランキングパルス42による割り込みがあった場合に即時実行状態となることはない。そして、この属性2のタスクは、属性1のタスクによって実行を中断された場合を除き、必ず各タスクのプログラムの最初から実行されるようになっている。本実施の形態では、タスクBおよびタスクCが属性2に属するものとし、このうち、タスクBの優先順位が2位、タスクCの優先順位が3位であるとして説明する。

【0024】本OSを図1に示した映像表示装置に適用する場合においては、垂直ブランキングパルス42による割り込みによって起動するタスクAは、例えば内部バス

10

20

30

40

50

アクセスを担当するタスクであり、タスクBおよびタスクCは、例えばキースキャンやOSDデータの管理を担当するタスクである。

【0025】RAM13は、CPU11内の上記したレジスタ群の内容を一時的に退避するためのスタック領域や、各種のバッファメモリ領域等のワークエリアとして用いられるもので、例えばDRAM (Dynamic Random Access Memory)によって構成される。

【0026】図4(a)はCPU11のレジスタ群を表し、同図(b)はROM12およびRAM13におけるメモリアドレスの割り当て状態を表すものである。この図の(a)に示したように、CPU11のレジスタ群には、プログラムカウンタ、インストラクションレジスタ、フラグレジスタ、およびその他の汎用レジスタ等が含まれている。また、この図の(b)に示したように、ROM12およびRAM13には、連続したメモリアドレスが割り当てられており、このうち、RAM13には、OS用スタック領域S0と、BC共用スタック領域SBCと、A用スタック領域SAと、他のRAM領域とが割り当てられている。ここで、OS用スタック領域S0は、OSがタスクA~CのいずれかにCPU11の占有権を開放する際にレジスタ群の内容を一時的に退避させる領域であり、A用スタック領域SAは、タスクAの実行中に外部からのより緊急度の高い割り込みが生じた場合や、自らサブルーチンコールを行った場合等に、その時点のCPUのレジスタ群の内容を一時退避するためのものである。また、BC共用スタック領域SBCは、タスクB(またはC)がOSもしくは他のタスクCにCPU11の占有権を開放する際にレジスタ群の内容を一時的に退避させる領域である。このBC共用スタック領域SBCは、タスクBとタスクCとにより共用される領域であるが、後述するように、OSは、BC共用スタック領域SBCが2つのタスクB、Cによって同時に使用されないように各タスクの状態制御を行う構成になっている。この点が、本発明の特徴の1つをなしている。

【0027】また、図4(b)における他のRAM領域には、タスクA~Cのそれぞれについて、スタックポインタ(SP)が指すスタック領域のメモリアドレスを格納するためのスタックポインタ用バッファ(A用SPバッファ、B用SPバッファ、およびC用SPバッファ)が設けられている。なお、本図では、これらのバッファの図示を省略している。

【0028】図5は各タスクにおける状態遷移を表すものである。この図に示したように、各タスクは、実行(RUN)、レディ(READY)、SPLレディ(SPECIAL READY)、待ち(WAIT)、および休眠(DORMANT)の各状態を取り得るようになっている。ここで、実行状態とはタスクが実行中である場合をいい、休眠状態とは、タスクがまだ起動されていない状態、またはタスクが終了した状態をいう。待ち状

態とは、そのタスクを実行するための条件が整うのを待っている状態であり、具体的には、指定された時間の経過を待っている状態と、タスクが指定したセマフォ(Semaphore)が資源を獲得するのを待っている状態とがある。レディ状態とは、自タスク側の実行の準備は整っているが、他のタスクが実行中である状態をいう。SPLレディ状態とは、垂直ブランキングパルス42による属性1のタスク(ここではタスクA)の実行開始によって実行が中断されているタスクの状態をいい、レディ状態の中の特異な状態である。このSPLレディ状態にあるタスクは、属性1のタスクの実行が終了したときは、設定された優先順位に係わりなく(すなわち、他の優先順位の高いタスクがレディ状態にあったとしても)実行が再開されるタスクである。このSPLレディ状態を設けた点が本発明の特徴の一つをなしている。

【0029】次に、図5および図6を参照して、以上のような構成の主制御部31におけるCPU11の動作を説明する。なお、図6は、タスクA~Cのいずれかのタスク(以下の説明では自タスクと表現する。)がOSの制御の下でどのようにして自己の状態を遷移させるかを表すものである。ここでは、当初のタスク状態が休眠状態となっているものとして説明する。

【0030】図5において、自タスクが休眠状態にある場合において、自タスクを起動させるためのスタート命令が他のタスクからOSになされると(図6ステップS101; Y)、自タスクは直ちにレディ状態に遷移する(ステップS102)。ここで、自タスクが即時実行可能状態であって(ステップS103; Y)、かつ、その時点でレディ状態となっているタスクの中で自タスクの優先順位が最も高く(ステップS104; Y)、しかもその時点でSPLレディ状態にある他のタスクが存在しないときには(ステップS105; Y)、自タスクは、直ちに実行状態へと遷移する(ステップS106)。また、自タスクの優先順位が最も高いが(ステップS104; Y)、その時点でSPLレディ状態にある他のタスクが存在するときには(ステップS105; N)、自タスクは実行状態に遷移せず、そのままレディ状態を維持する。

【0031】一方、ステップS101においてスタート命令があってもレディ状態に移った場合でも、自タスクが即時実行可能状態でないときには(ステップS103; N)、自タスクはそのままレディ状態を維持するが、この状態で自タスクを待ち状態にするためのウェイト命令が他のタスクからOSになされると(ステップS107; Y)、自タスクは待ち状態に遷移する(ステップS108)。さらに、この状態で他のタスクから待ち状態解除命令がなされた場合、あるいは、所定時間の経過により自動的に待ち状態解除が行われるような設定になっていた場合にはその設定された時間の経過により、自タスクは再びレディ状態に遷移する(ステップS10

2)。

【0032】図6のステップS106において実行状態に遷移したのちは、次のように動作する。すなわち、自タスクを待ち状態にするためのウェイト命令を自タスクからOSに対して行うことにより(ステップS110; Y)、自タスクは実行状態から待ち状態へと遷移する(ステップS108)。また、属性1のタスクが起動して自タスクの実行が中断されたときは(ステップS111; Y)、自タスクはSPLレディ状態へと遷移する(ステップS114)。また、自タスクを休眠状態にするための終了命令を自タスクからOSに対して行うことにより(ステップS112; Y)、自タスクは実行状態から休眠状態へと遷移する(ステップS113)。それ以外の場合は(ステップS110; N, ステップS111; N, ステップS112; N)、その実行状態を維持する。

【0033】ステップS114においてSPLレディ状態に遷移したのちは、その状態遷移の原因となった属性1のタスクの実行状態の終了を待って(ステップS115; Y)、自タスクは実行状態となり、実行を再開する(ステップS106)。このように、SPLレディ状態にある自タスクは、属性1のタスクの実行状態の終了時点において自分よりも優先順位の高い他のレディ状態のタスクが存在していたとしても、それに優先して自らが実行状態となることができる。

【0034】次に、図7～図15を参照して、本OSによるタスクA～Cの制御に係る具体例を説明する。ここで、図7はOSおよびタスクA～Cにおける時刻経過に伴うCPUの占有状況の一例を表し、図8、図14および図15は、それぞれ、図7の状況C1～C2を詳細に表すものである。また、図9～図13は、図8の状況C1におけるCPU11および各メモリの状態変化を表すものである。

【0035】本実施の形態では、タスクA(図7(d))は垂直ブランキングパルス42(同図(c))に同期して直ちに起動して実行状態となり、タスクB(同図(e))は、CPU11の内部で一定周期で発生するOS基準割込信号2(同図(a))のタイミングで起動をかけられるものとする。また、タスクC(同図(f))は、CPU11の内部で一定周期で発生する他のOS基準割込信号1(図7(b))の2周期ごとのタイミングで起動をかけられるようになっていたものとする。

【0036】まず、図8および図9～図13を参照して、図7の状況1における動作を詳細に説明する。

【0037】図8に示したように、時刻t1以前においては、OSにCPU11の占有権があり、タスクA～Cのすべてが待ち状態である。いずれかのタスクがレディ状態になるのをOSが待っている状態である。この状態では、図9に示したように、CPU11のプログラムカ

ウンタはOS自身のルーチンのうちのいずれかのアドレスを指しており、OS用スタック領域S0が使用されている。スタックポインタSPは、OS用スタック領域S0の使用領域のうちのいずれかのアドレスを指している。このとき、A用SPバッファには、A用スタック領域SAのスタートアドレスがセットされ、B用SPバッファおよびC用SPバッファには、共にBC共用スタック領域SBCのスタートアドレスがセットされている。

【0038】ここで、時刻t1においてOS基準割込信号による割込みがあると、タスクCは直ちにレディ状態となる。OSはCPU11の占有権をどのタスクに渡すべきかを判断する。この時点では、優先順位の最も高いタスクAおよび次に優先順位の高いタスクBは共に待ち状態になっているため、図10に示したように、OSはスタックポインタSPの示すアドレスをBC共用スタック領域SBCのスタートアドレスにセットすると共に、プログラムカウンタをタスクCの最初のアドレスにセットし、CPU11の占有権をタスクCに渡す。これにより、タスクCは実行状態となる。このとき、A用SPバッファには、A用スタック領域SAのスタートアドレスがセットされ、B用SPバッファおよびC用SPバッファには、共にBC共用スタック領域SBCのスタートアドレスがセットされている。

【0039】時刻t1でタスクCが実行状態となったのち、このタスクCよりも優先順位の高いタスクBについて設定されていた起動待ちタイマがタイムアップしたとする。従来の方法では、ここでタスクBがタスクCを中断して実行状態となっていたが、本実施の形態では、図8(c)に示したように、タスクBはタスクCを中断することなく、単に、待ち状態からレディ状態に遷移するのみである。

【0040】さらに、時間が経過して、時刻t2において垂直ブランキングパルス42による割込みがかけられると、OSは、図11に示したように、この時点でのプログラムカウンタ等のCPU11のレジスタ群の内容をBC共用スタック領域SBCに退避(プッシュ)すると共に、このときのスタックポインタSPの値をB用SPバッファおよびC用SPバッファに格納する。これにより、B用SPバッファおよびC用SPバッファの内容は、BC共用スタック領域SBCのうちの使用されている領域の最終アドレスとなる。そして、OSは、A用SPバッファに格納されているスタックポインタSPの値をA用スタック領域SAのスタートアドレスとし、さらに、プログラムカウンタをタスクAの先頭アドレスにセットする。これにより、タスクAはレディ状態を経て実行状態に遷移し、タスクCは実行状態からSPLレディ状態に遷移する。なお、タスクBはそのままレディ状態を維持する。

【0041】タスクAは、時刻t3において必要な処理が終了すると、自らを待ち状態とするためのウェイト命

令をOSに出し、OSにCPU11を開放して自らは待ち状態となる。このときのタスクBの状態はレディ状態であり、タスクCの状態はSPLレディ状態となっている。このため、OSは、より優先順位の高いタスクBを差し置いてタスクCを優先して実行状態に遷移させ、タスクAにより中断されていた処理を再開させる。具体的には、OSは、BC共用スタック領域SBCに退避されていた各種のレジスタデータをCPU11のレジスタ群に復帰（ポップ）させる。これにより、図12に示したように、プログラムカウンタはタスクCの再開アドレスにセットされる。また、このとき、B用SPバッファおよびC用SPバッファに格納されているアドレス（この場合、タスクCがタスクAによって実行を中断された時点の最終アドレス）をスタックポインタSPに復帰させ、各バッファはBC共用スタック領域SBCの先頭アドレスに戻される。この点が本発明の特徴の1つをなしている。なお、このときのA用SPバッファの内容はA用スタック領域SAのスタートアドレスとなり、B用SPバッファおよびC用SPバッファの内容は、BC共用スタック領域SBCのうちの使用されている領域の先頭アドレスとなり、また、スタックポインタSPはBC共用スタック領域SBC内におけるタスクCが使用した最終アドレスとなる。

【0042】実行を再開したタスクCは、時刻t4において必要な処理が終了すると、自らを待ち状態にすることを求めるウェイト命令をOSに出してCPU11を開放する。すると、OSは、この時点で唯一レディ状態となっているタスクBを実行状態に遷移させる。具体的には、図13に示したように、プログラムカウンタをタスクBの先頭アドレスにセットすると共に、上記したようにBC共用スタック領域SBCの先頭アドレスに戻されたB用SPバッファおよびC用SPバッファの値を、スタックポインタSPにコピーする。これにより、これ以降、BC共用スタック領域SBCはタスクBによって使用可能となる。なお、このとき、タスクAは待ち状態を維持し、A用SPバッファの内容はA用スタック領域SAのスタートアドレスとなっている。

【0043】実行状態となったタスクBは、時刻t5において必要な処理を終了させると、自らを待ち状態にすることを求めるウェイト命令をOSに出してCPU11をOSに開放する。これにより、状況1における処理が終わる。

【0044】このように、あるタスクが実行状態である場合には、属性1以外の属性をもつより優先順位の高いタスクから割込要求がなされたとしても、実行状態のタスクはこのタスクにCPU11の占有権を渡すことはなく、実行途中のタスクが終了するまで、その優先順位の高いタスクは待つこととなる。但し、属性1のタスクからの割込要求についてはこの限りでなく、要求に応じて直ちにCPU11の占有権が与えられることとなる。ま

た、レディ状態にあるタスクとSPLレディ状態にあるタスクとが競合したときは、優先順位の上下に係わりなく、SPLレディ状態にあるタスクにCPU11の占有権が与えられることとなる。

【0045】次に、図14を参照して、図7における状況2を説明する。

【0046】ここでは、時刻t6において、OS基準割込信号による割込みが行われると同時に、タスクB用の起動待ちタイマがタイムアップしたとする。この場合、タスクBおよびタスクCの双方がレディ状態となっているが、タスクCよりもタスクBの方が優先順位が高く設定されている。このため、タスクBが実行状態となってCPU11を占有する。一方、タスクCはレディ状態となって、タスクBが実行を終了してCPU11を開放するのを待つ。時刻t7においてタスクBによる必要な処理が終了すると、タスクBは休眠状態になる一方、タスクCは実行状態になる。そして、タスクCは、必要な処理を行ったのち、時刻t8においてCPU11をOSに開放し、実行状態から待ち状態に遷移する。

【0047】このように、タスクBとタスクCとが共にレディ状態にあるときに、起動条件が競合したときは、原則通り、優先順位に従って、より高い優先順位のタスクに対してCPU11の占有権が与えられることとなる。

【0048】以上のように、本実施の形態に係るマルチタスク制御方法では、タスクBとタスクCとが共にレディ状態にあって起動要求が競合したときは、原則通り、優先順位に従って、より高い優先順位のタスクに対してCPU11の占有権が与えられるが、属性2のタスクBまたはCの実行中に属性1のタスクAから割込要求があったときは、直ちにCPU11の占有権をタスクAに与える一方、タスクB（またはC）の実行中にタスクC（またはB）からの割込み要求があったときは、そのタスクの優先順位の如何に係わらず、その実行中のタスクB（またはC）の終了を待ってCPU11の占有権を開放するようにしたので、必要なタスク処理のみを限定的にリアルタイム動作させ、さほどリアルタイム性を必要としないタスクについてはいわば擬似的なリアルタイム動作をさせることができる。このため、不要な機能についての処理プログラム部分を削減することができ、OS自体のプログラム構成をシンプルにできる。また、属性1以外のタスク（BまたはC）については、同時に同一のスタック領域を使用することがなくなるので、スタック領域を複数のタスク（BおよびC）によって共用することが可能となる。

【0049】また、レディ状態にあるタスクとSPLレディ状態にあるタスクとが競合したときは、優先順位の上下に係わりなく、必ず、SPLレディ状態にあるタスクにCPU11の占有権が与えられるようにしたので、タスクAによって実行を中断されたタスクBまたはC

は、タスクAの実行終了後、確実に実行を再開することができる。

【0050】なお、以上のような動作を行うOSは、例えば次のようにして所定の記録媒体に記録可能である。すなわち、まず、例えばC言語を用いてOS本体部および各タスクに係るプログラムを作成する。次に、これらのプログラムをコンパイラによりそれぞれコンパイルして、アセンブリ言語で記述されたプログラムに変換する。次に、これらのアセンブリ言語で記述されたプログラムをアセンブラを用いてそれぞれマシン語に変換すると共に、これらの複数のマシン語プログラムをリンカを用いて統合して1つのプログラムとする。こうして作成されたOS全体のプログラムは、例えばフロッピーディスク、マスクROM、EPROM（消去・再書込可能ROM）、またはOTP（ワンタイムPROM）等へ書き込まれ、あるいは、マイクロコンピュータのチップ自体に組み込まれて、一般に提供可能となる。

【0051】以上、いくつかの実施の形態を挙げて本発明を説明したが、本発明はこれらの実施の形態に限定されず、種々変形可能である。例えば、上記実施の形態では、属性2のタスクはタスクBおよびタスクCの2つとしたが、それ以上の数のタスクに対しても適用可能である。また上記の実施の形態では、タスクAは垂直ブランピングパルス42による割込みによって起動することとしたが、他の割込み信号によって起動されるようにしてもよい。また、本実施の形態では、電子機器として映像表示装置を例にとって説明したが、他の電子機器にも同様に適用可能である。

#### 【0052】

【発明の効果】以上説明したように本発明に係るマルチタスク制御方法によれば、即時実行が要求される特定のタスクに対してのみ限定的にリアルタイム制御を行い、それ以外のタスクに対しては擬似的なリアルタイム制御を行うようにしたので、すべてのタスクに対して平等にリアルタイム制御を行うようにした従来の汎用マルチタスク制御方法に比べて、それを実現するためのプログラムのサイズが小さくて済み、プログラム格納用のメモリ容量を削減することができる。しかも、上記した特定のタスク以外のタスクについてはスタック領域を共用化することができるので、スタックメモリの容量をも削減することができる。したがって、マルチタスク制御に必要なメモリ容量を削減することができる。さらに、その特定のタスク以外のタスクに関してはリアルタイム性をそれほど要求しないシステムにとっては、メモリ資源が無駄に消費されるのを防止することができる。また、本発明

のマルチタスク制御方法によれば、その機能構成が単純であるので、それを実現するためのプログラムの殆どを例えばC言語等によって開発することができ、これにより、異なるマイクロコンピュータシステムへの移植が容易となる。

#### 【図面の簡単な説明】

【図1】本発明の一実施の形態に係るマルチタスク制御方法が適用される電子機器としての映像表示装置の概略構成を表すブロック図である。

10 【図2】図1の映像表示装置における主制御部の概略構成を表すブロック図である。

【図3】図2のCPUの動作を制御するオペレーティングシステムと各タスクとの関係を表す図である。

【図4】図2のCPU内のレジスタ群とROMおよびRAM内のメモリアドレスとの関係を表す図である。

【図5】タスクの状態遷移の様子を表す図である。

【図6】タスクの状態遷移を説明するための流れ図である。

20 【図7】各タスクの動作タイミングの一例を表すタイミング図である。

【図8】図7における一部の動作タイミングを詳細に表す図である。

【図9】図8に示した動作を行っているCPUおよびRAMのある時点での状態を表す図である。

【図10】図8に示した動作を行っているCPUおよびRAMの他の状態を表す図である。

【図11】図8に示した動作を行っているCPUおよびRAMのさらに他の状態を表す図である。

30 【図12】図8に示した動作を行っているCPUおよびRAMのさらに他の状態を表す図である。

【図13】図8に示した動作を行っているCPUおよびRAMのさらに他の状態を表す図である。

【図14】図7における他の部分の動作タイミングを詳細に表す図である。

【図15】従来のマルチタスク制御方法の一例を説明するためのタイミング図である。

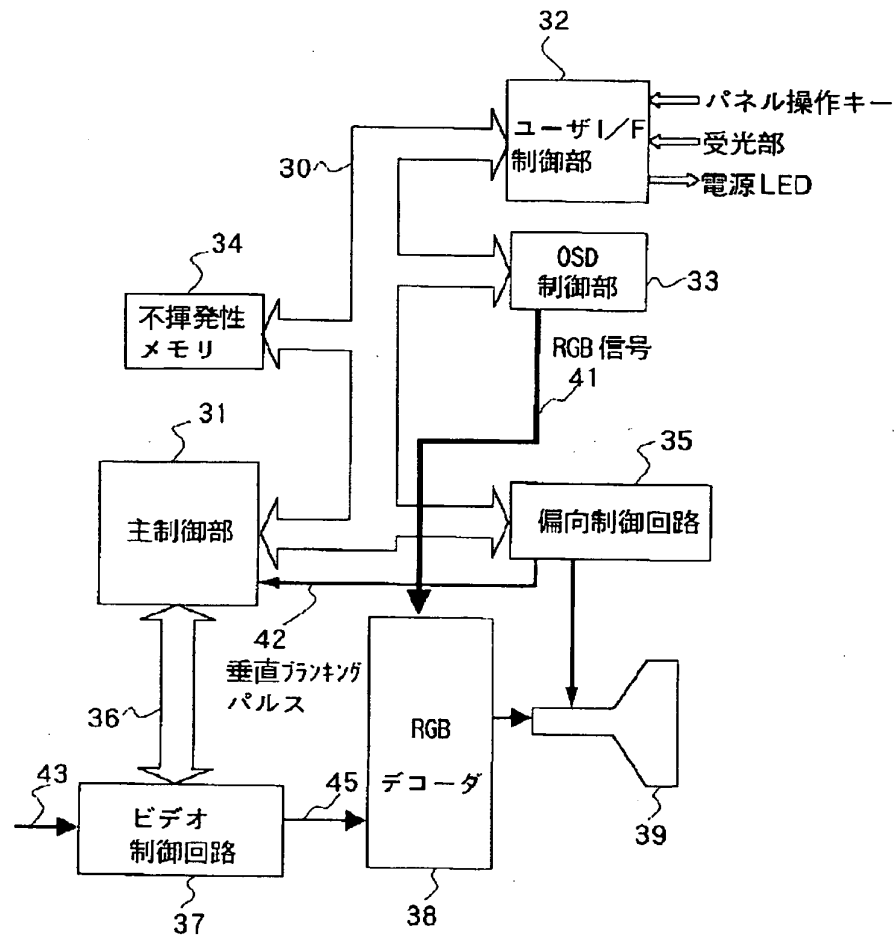
【図16】従来のマルチタスク制御方法におけるスタック領域の構成例を表す図である。

#### 【符号の説明】

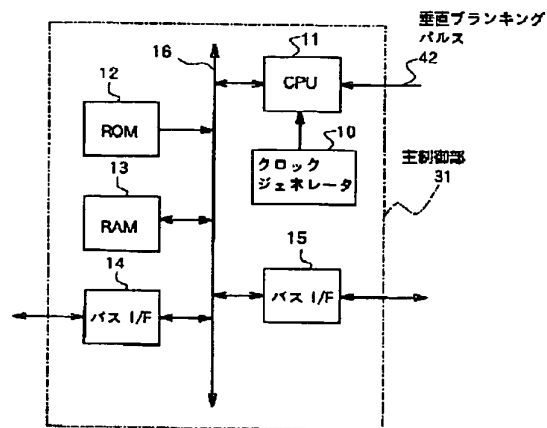
40 1、2…OS基準割込信号、11…CPU、12…ROM、13…RAM、31…主制御部、42…垂直ブランピングパルス、SA…A用スタック領域、SBC…BC共用スタック領域、S0…OS用スタック領域、A～C…タスク



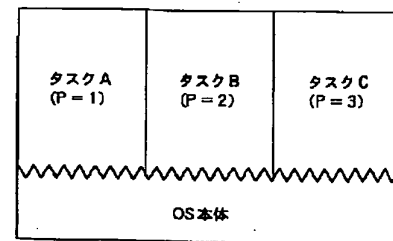
【図1】



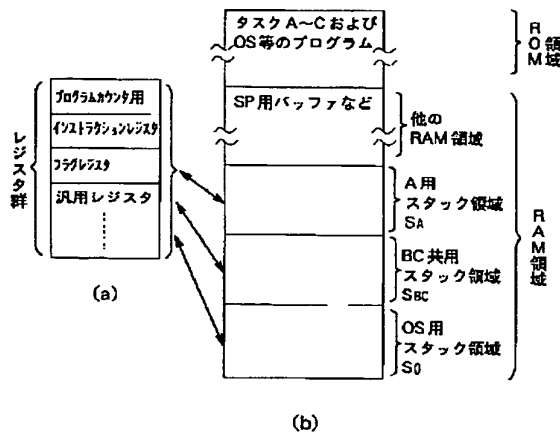
【図2】



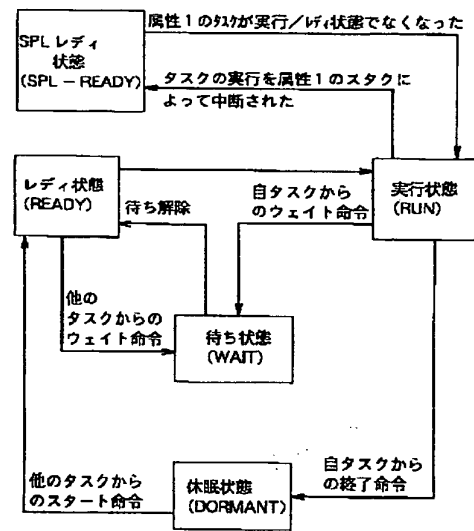
【図3】



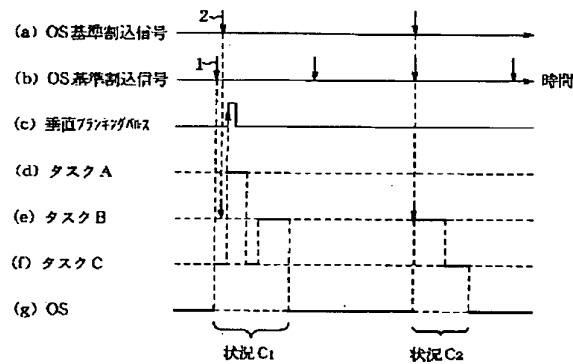
【図4】



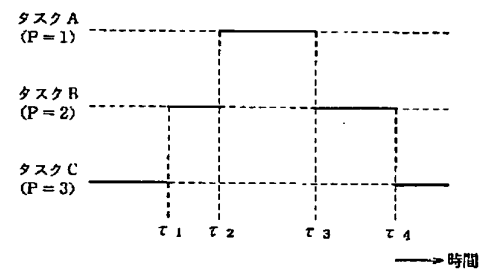
【図5】



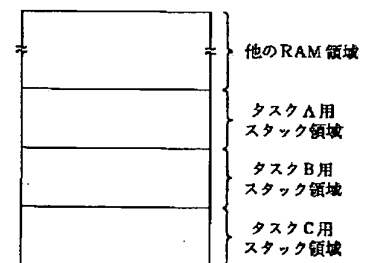
【図7】



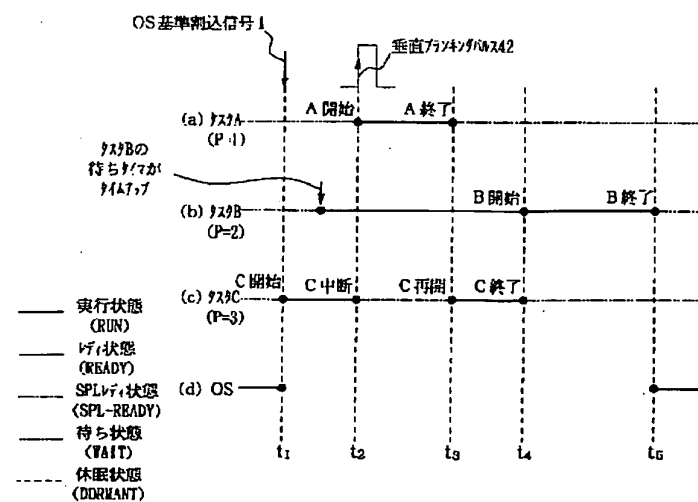
【図15】



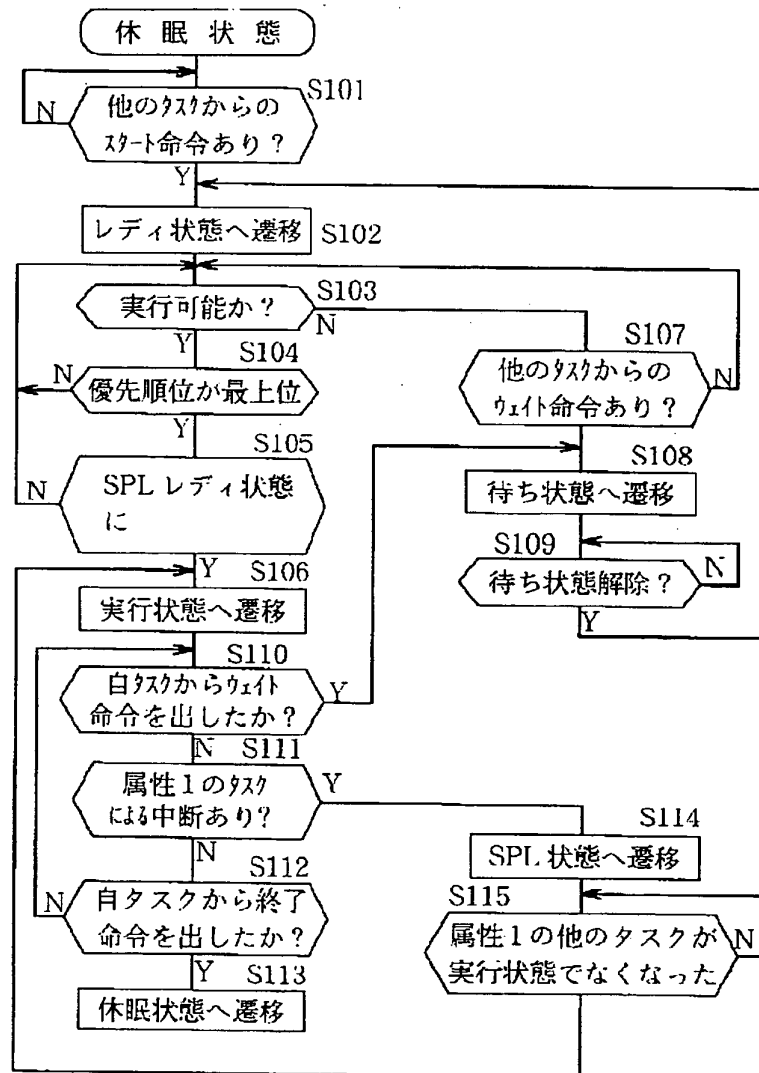
【図16】



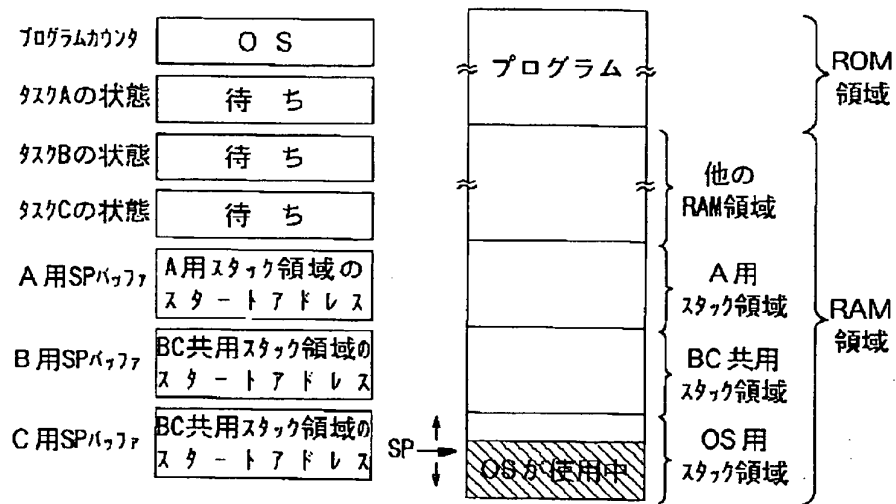
【図8】



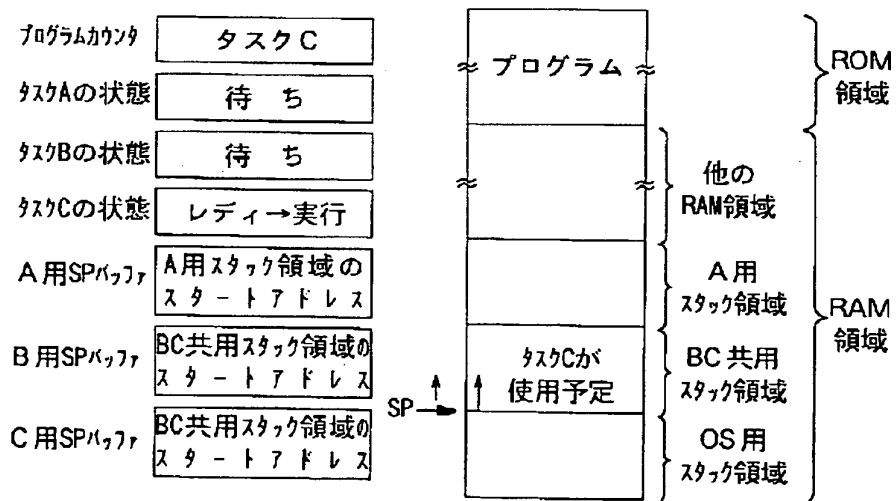
【図6】



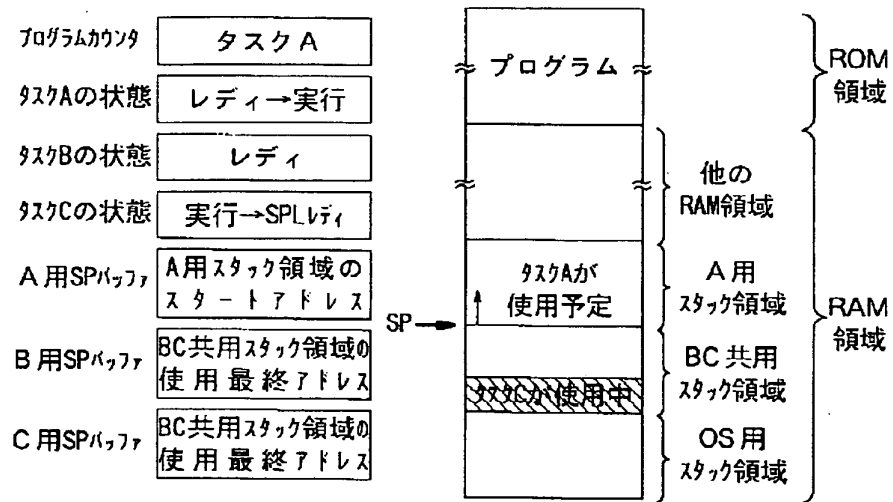
【図9】



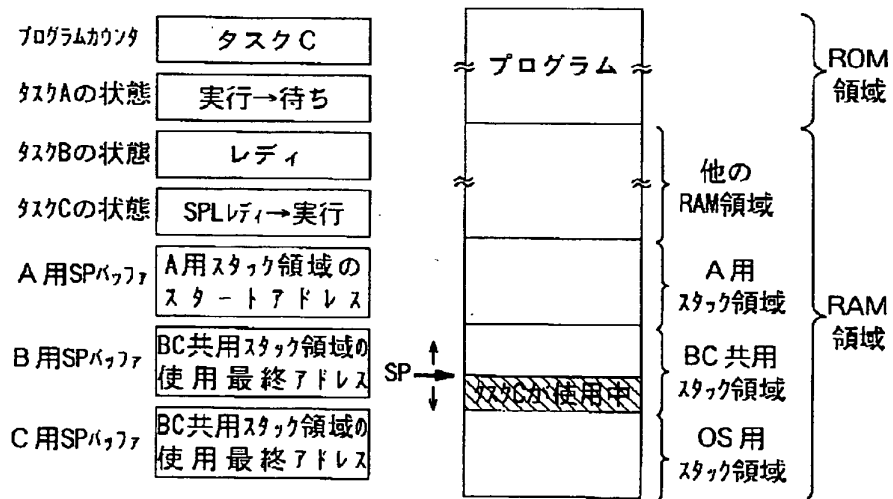
【図10】



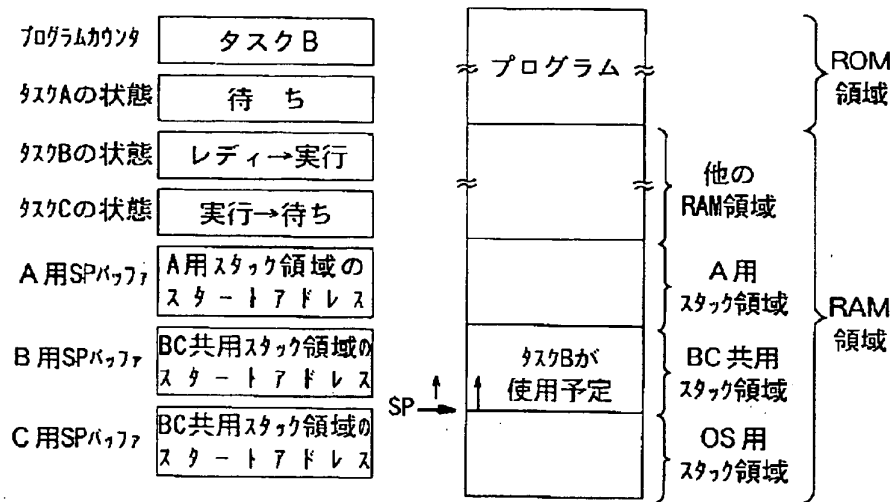
【図11】



【図12】



【図13】



【図14】

